An In-depth Review on the AV1 Intra Prediction

Marcel M. Corrêa (Ph.D. student)**,**
Daniel M. Palomino, Guilherme R. Corrêa (Co-advisors)**,**
Luciano V. Agostini (Advisor)

Video Technology Research Group (ViTech)
Programa de Pós-Graduação em Computação (PPGC)
Universidade Federal de Pelotas (UFPel)

{mmcorrea, dpalomino, gcorrea, agostini}@inf.ufpel.edu.br

Abstract. *The AOMedia Video 1 (AV1) is an open-source and royalty-free video coding format, which was developed by the Alliance for Open Media (AOM) industry consortium and released in June 2018. The main goal of AV1 development was to achieve substantial compression gain over state-of-the-art codecs, while keeping a practical decoding complexity, hardware feasibility and its open and free status. This monograph provides a brief technical overview on the AV1 block partitioning and transform coding, and an in-depth technical review of its intra prediction tools, along with a brief presentation of the VP9 and HEVC intra prediction tools.*

1. Introduction

The continuous growth in consumption of digital videos over the internet, including services such as video-on-demand, video sharing on social networks, as well as high definition video conferences (even on mobile devices), are reaching the limits of available bandwidth of the telecommunication infrastructures. To address this situation, standardization bodies have been developing video coding standards during the last years. The most recent ISO/IEC and ITU-T standard is the High Efficiency Video Coding (HEVC) [Sullivan 2012; ISO/IEC 2013], which has many of its techniques protected by patents.

A key factor in the success of the internet is that its core technologies are open and freely implementable, and digital videos certainly are a central part of the internet experience nowadays. According to Cisco Systems, digital videos are consuming the majority of all the internet traffic and are expected to consume even more in the near future [Cisco Systems 2017], which makes open-source and royalty-free video formats highly desirable.

Based on that, Google LLC purchased the company On2 Technologies, which originally developed the VP8 video format [Bankoski 2011], and released the VP8 codec code under a BSD-like license and the bitstream specification under an irrevocable free patent license. This episode marked the start of the WebM project, as an effort to develop open media formats, later resulting in the release of the VP9 video format [Mukherjee 2013; Grange 2013].

Although Google LLC has been using VP9 successfully on YouTube, it is considered to be not enough to handle new tendencies, such as Ultra High Definition (UHD) 4K resolution which is four times larger than 1080p, UHD 8K resolution which is four times larger than 4K, High Dynamic Range (HDR) which has 25% to 50% more data than Standard Dynamic Range (SDR) and, also, higher frame rates. These tendencies result in an expressive increase of data to be compressed.

As Google LLC was working on a successor for VP9, called VP10 [Mukherjee 2015], and Cisco Systems and Mozilla Corporation were working on their own video formats, respectively called Thor [Bjøntegaard 2016] and Daala [Valin 2016], an industry consortium, called Alliance for Open Media (AOM), was formed. This alliance was made mainly because the paradigm of developing the technology first and solving the licensing deals later was not working for the majority of the companies involved, as Rosenberg (2015) explained:

> Unfortunately, the patent licensing situation for H.265 [HEVC] has recently taken a turn for the worse. Two distinct patent licensing pools have formed so far, and many license holders are not represented in either. There is just one license pool for H.264. The total costs to license H.265 from these two pools is up to sixteen times more expensive than H.264, per unit. H.264 had an upper bound on yearly licensing costs, whereas H.265 has no such upper limit.
> […]
> We [Cisco Systems] also hired patent lawyers and consultants familiar with this technology area. We created a new codec [Thor] development process which would allow us to work through the long list of patents in this space, and continually evolve our codec to work around or avoid those patents.

As result, the AOMedia Video 1 (AV1) [Chen 2018; Rivaz 2018] was released in June 2018 as the VP9 successor, to be a next-generation, open-source and royalty-free video format developed based on many elements of VP10, Thor and Daala.

This monograph provides a brief technical overview on the AV1 block partitioning and transform coding, and an in-depth technical review of its intra prediction tools, along with a brief presentation of the VP9 and HEVC intra prediction tools.

2. AV1 Block Structure and Transform Coding

In AV1, a video frame is partitioned into superblocks (SBs) of size 128x128 or 64x64 pixels, according to a bitstream flag. Then, SBs are predicted in raster order within the frame. To deliver a locally optimal prediction for each SB, the encoder can further divide each SB using a 10-way partition tree structure [Chen 2018], as illustrated in Figure 1. In this figure, blue blocks are final partition modes, but all four partitions of the white block can be recursively divided based on the same 10-way tree structure, down to 4x4 blocks, which is the smallest supported block size. The numbers inside each partition indicate the raster prediction order within the block.

Each final partition can be either intra or inter predicted, or even by a compound inter-intra mode, which combines a single-reference inter prediction and an intra prediction separated by a smoothing function [Chen 2018].

Specifically, for intra prediction, the process is not invoked at block level, but at transform block level instead. That is, when the transform size is smaller than the block size, the prediction is invoked multiple times in raster order within the block [Rivaz 2018], thus allowing the prediction of a transform block to use the previously predicted and reconstructed transform block as a better reference. The AV1 specifies many square and non-square sizes for transform blocks, from 4x4 to 64x64, as listed in Table 1. In this table, *TxSize* represents how a given transform size is signaled in the bitstream.

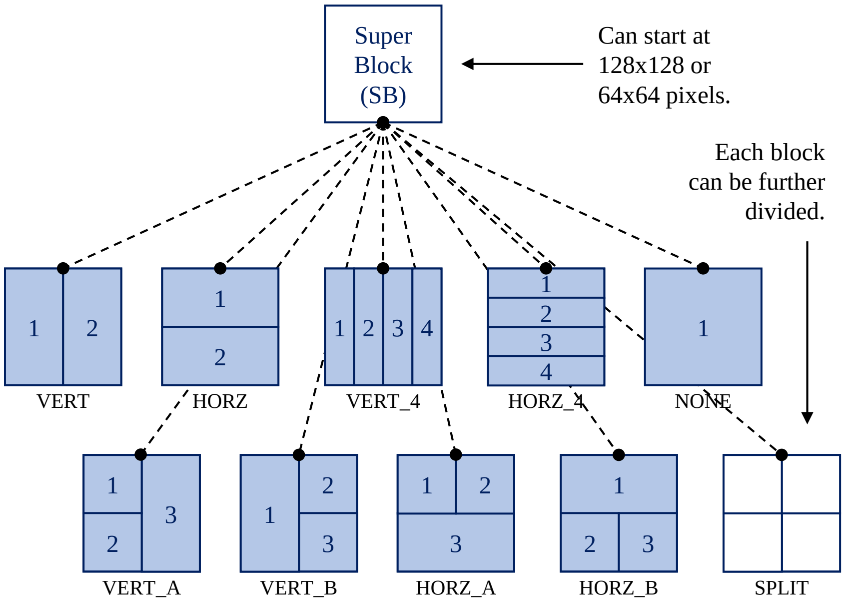

Figure 1. AV1 10-way block partition tree structure.

Table 1. AV1 supported transform sizes [Rivaz 2018].

TxSize	Transform Size
0	4x4
1	8x8
2	16x16
3	32x32
4	64x64
5	4x8
6	8x4
7	8x16
8	16x8
9	16x32
10	32x16
11	32x64
12	64x32
13	4x16
14	16x4
15	8x32
16	32x8
17	16x64
18	64x16

A very rich set of transform kernels is defined for both inter and intra blocks. The full set consists of 16 combinations of vertical and horizontal Discrete Cosine Transform (DCT), Asymmetrical Discrete Sine Transform (ADST), flipADST and Identity Transform (IDTX).

Intra prediction tends to be less accurate for predicted samples further away from the reference samples, resulting in a residue with higher concentration on the bottom right corner. For these cases, the asymmetrical transforms, such as the ADST and flipADST (the ADST applied in reverse order), are the most appropriate [Parker 2016].

Also, the IDTX, which skips the transform in the applied direction, shows good results for coding residue with very sharp edges. Moreover, the specific combination of IDTX vertical and IDTX horizontal results in a full transform skip, which is very effective for Screen Content Coding (SCC) [Parker 2016].

3. AV1 Intra Prediction

As explained in section 2, although AV1 supports many different block sizes ranging from 128x128 to 4x4 due to its flexible partition structure, only partitions of size 64x64 or smaller are supported by the intra prediction modes (predictors), according to the 19 transform sizes listed in Table 1.

AV1 supports a variety of non-directional predictors that consider gradients, spatial correlation of samples and coherence of luminance and chrominance planes. These modes are: DC, Paeth, Smooth, Smooth Vertical, Smooth Horizontal, Recursive-based-filtering (modes 0 to 4) and Chroma-from-luma (CfL) [Trudeau 2018]. There are also two modes particularly efficient for Screen Content Coding, namely the Intra Block Copy [Li 2018] and Palette [Guo 2014] modes. Finally, AV1 also supports many different directional predictors, corresponding to angles between 36 and 212 degrees, aiming at several varieties of spatial redundancies in directional textures.

Generally, the intra prediction of a single block, referred as a 2D array called *Pred* of size *width* by *height*, requires reference samples from previously reconstructed blocks located on the left and above the current block. The number of reference samples needed for each reference array, referred as *AboveRow* (row of samples above the current block) and *LeftCol* (column of samples to the left of the current block), is equal to *width*+*height*+1, as illustrated in Figure 2.

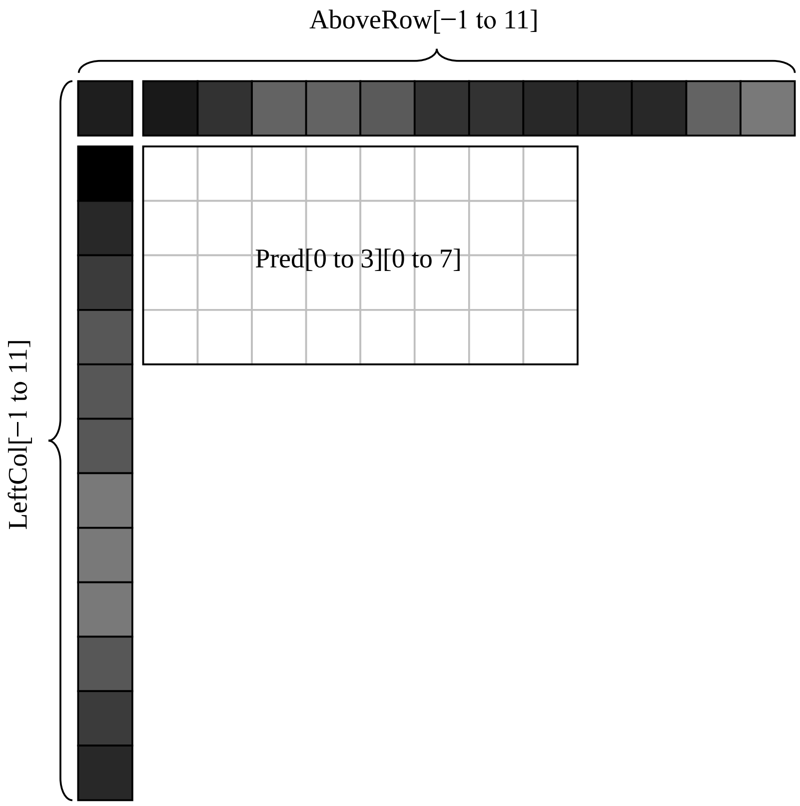

Figure 2. Example of a 4x8 block, which needs a total of 25 reference samples from previously encoded adjacent blocks. The position -1 of both reference arrays always share the same value.

The following subsections present detailed information about each intra prediction mode supported by the AV1.

3.1. DC Mode

This well-known predictor is also found in many older video formats, although this one was based on the VP9 version, which also considers the availability of reference samples. That is, because of the block processing order or data dependency break introduced by parallel encoding tools, the top and/or left neighboring blocks may not be available to be used as reference in intra prediction.

As described in the algorithm from Figure 3, when both the top and left reference samples are available, the DC algorithm generates a solid surface, represented by the average between these references. However, if one of the reference samples array is available and the other is not, then only references from the available array are used to compute the average and the rounding is adjusted accordingly. Lastly, if there are no references available, then the surface is simply represented by $1 << (bitdepth - 1)$, where *bitdepth* in AV1 can either be 8, 10 or 12 bits. All four cases are exemplified in Figure 4.

```
dc = 0
FOR i in 0 to height - 1:
  dc = dc + LeftCol[i]
END FOR
FOR j in 0 to width - 1:
  dc = dc + AboveRow[j]
END FOR
dc = dc + (width + height) / 2
dc = dc / (width + height)
FOR i in 0 to height - 1:
  FOR j in 0 to width - 1:
    Pred[i][j] = dc
  END FOR
END FOR
```

Figure 3. DC algorithm for when both the top and left neighboring blocks are available.

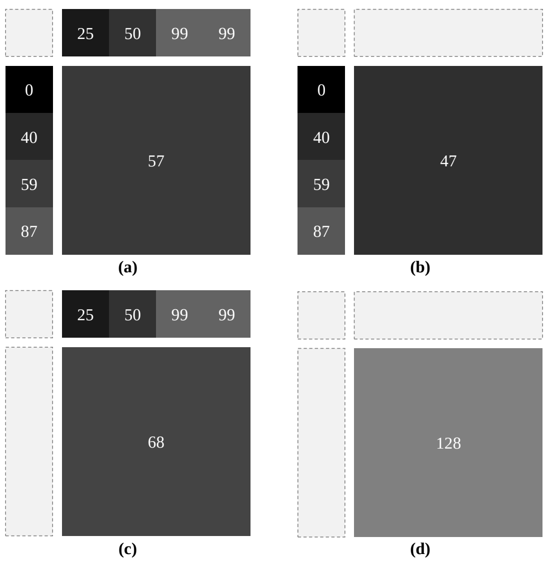

Figure 4. Example of DC-predicted blocks of size 4x4 when: (a) all references are available, (b) only left references are available, (c) only top references are available, and (d) all references are unavailable. Omitted reference samples are not used in the prediction process.

3.2. Paeth Mode

The Paeth predictor is a novel algorithm included in the AV1 format, based on the VP9 True-motion (TM) mode, which generates a fairly smooth surface as the predicted block by using only exact copies of reference samples.

As described in the algorithm from Figure 5, for each position of the predicted block, the algorithm compares three reference samples: the vertically aligned, the horizontally aligned, and the top-left reference. The predicted sample is a copy of the reference which leads to the lowest gradient [Chen 2018]. An example of a Paeth prediction is shown in Figure 6, where the function *ABS* returns the absolute value of the argument passed.

```
FOR i in 0 to height – 1:
  FOR j in 0 to width – 1:
    base = LeftCol[i] + AboveRow[j] – AboveRow[-1]
    pLeft    = ABS(base – LeftCol[i])
    pTop     = ABS(base – AboveRow[j])
    pTopLeft = ABS(base – AboveRow[-1])

    IF pLeft <= pTop AND pLeft <= pTopLeft:
      Pred[i][j] = LeftCol[i]
    ELSE IF pTop <= pTopLeft:
      Pred[i][j] = AboveRow[j]
    ELSE
      Pred[i][j] = AboveRow[-1]
    END IF
  END FOR
END FOR
```

Figure 5. Paeth algorithm describing how intermediate values are computed and then compared in order to select the optimal predicted sample.

30	25	50	99	99	90	50	50	40
0	0	30	99	99	90	30	30	0
40	40	50	99	99	90	50	50	40
59	59	59	99	99	90	59	59	59
87	87	87	99	99	90	87	87	87

Figure 6. Example of a Paeth predicted block of size 8x4. All predicted samples are copies of one of the three references samples compared.

3.3. Smooth, Smooth Vertical and Smooth Horizontal Modes

These three modes use linear interpolation to generate very smooth surfaces composed by filtered samples from the reference arrays *LeftCol* and *AboveRow*.

The Smooth Vertical mode interpolates predicted samples using references *AboveRow*[0 to *width*−1] and the bottommost reference *LeftCol*[*height*−1]. The Smooth Horizontal mode is an analogue process that uses references *LeftCol*[0 to *height*−1] and the rightmost reference *AboveRow*[*width*−1]. Lastly, the Smooth mode generates an average of the Smooth Vertical and Smooth Horizontal predictions.

The interpolation algorithms are described in Figures 7 and 8, where *SmoothCoefficients* is a constant array for the interpolation coefficients, according to the block *height* for the Smooth Vertical mode and block *width* for the Smooth Horizontal mode, as listed in Table 2. Examples of predicted blocks for each of the three modes are illustrated in Figure 9.

```
FOR i in 0 to height – 1:
  FOR j in 0 to width – 1:
    a =  SmoothCoefficients[i] * AboveRow[j]
    b = (256 – SmoothCoefficients[i]) * LeftCol[height-1]
    Pred[i][j] = (a+b+128)/256
  END FOR
END FOR
```

Figure 7. Smooth Vertical algorithm describing how the linear interpolation happens between references of the *AboveRow* and the bottommost reference of *LeftCol*.

```
FOR i in 0 to height – 1:
  FOR j in 0 to width – 1:
    a =  SmoothCoefficients[j] * LeftCol[i]
    b = (256 – SmoothCoefficients[j]) * AboveRow[width-1]
    Pred[i][j] = (a + b + 128) / 256
  END FOR
END FOR
```

Figure 8. Smooth Horizontal algorithm describing how the linear interpolation happens between references of the *LeftCol* and the rightmost reference of *AboveRow*.

Table 2. *SmoothCoefficients* array read by the Smooth Vertical and Smooth Horizontal modes.

Size	Coefficients
4	{255, 149, 85, 64}
8	{255, 197, 146, 105, 73, 50, 37, 32}
16	{255, 225, 196, 170, 145, 123, 102, 84, 68, 54, 43, 33, 26, 20, 17, 16}
32	{255, 240, 225, 210, 196, 182, 169, 157, 145, 133, 122, 111, 101, 92, 83, 74, 66, 59, 52, 45, 39, 34, 29, 25, 21, 17, 14, 12, 10, 9, 8, 8}
64	{255, 248, 240, 233, 225, 218, 210, 203, 196, 189, 182, 176, 169, 163, 156, 150, 144, 138, 133, 127, 121, 116, 111, 106, 101, 96, 91, 86, 82, 77, 73, 69, 65, 61, 57, 54, 50, 47, 44, 41, 38, 35, 32, 29, 27, 25, 22, 20, 18, 16, 15, 13, 12, 10, 9, 8, 7, 6, 6, 5, 5, 4, 4, 4}

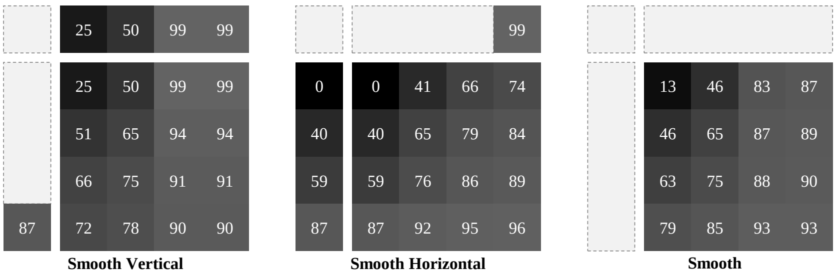

Figure 9. Examples of blocks of size 4x4 predicted by the Smooth Vertical, Smooth Horizontal and Smooth modes. Omitted reference samples are not required by the prediction mode.

3.4. Recursive-based-filtering Mode

This group of predictors is also a novelty of the AV1 format, specifically designed for luminance blocks, to mitigate decaying spatial correlation as positions in the predicted block get further away from the reference samples. It works by first breaking the current block to be predicted, regardless of size, into 4x2 subblocks (with seven reference samples adjacent to it). The subblocks not fully attached to real reference samples must use previously predicted values of adjacent subblocks as reference, meaning prediction is computed recursively among subblocks.

As specified in [Rivaz 2018], each 4x2 subblock is defined as an array *Block*, indexed in raster order, which means the first row has predicted samples indexed as *Block*[0 to 3] and the second row indexed as *Block*[4 to 7]. The reference samples for a given subblock are defined as an array *L*, indexed as follows: *L*[0] is the top-left adjacent sample, *L*[1 to 3] are the four top adjacent samples, and *L*[5 to 6] are the two left adjacent samples.

As described in the base algorithm from Figure 10, each predicted sample of a 4x2 subblock is the result of a different 7-tap filter between all the seven references. Each mode differs from each other only by the coefficients used in the filters. The *FilterCoefficients* is a constant 3D array for all the coefficients, as listed in Table 3.

Table 3. *FilterCoefficients* array read by the five Recursive-based-filtering modes.

Mode	*Block* index	Coefficients
0	0	{−6, 10, 0, 0, 0, 12, 0}
	1	{−5, 2, 10, 0, 0, 9, 0}
	2	{−3, 1, 1, 10, 0, 7, 0}
	3	{−3, 1, 1, 2, 10, 5, 0}
	4	{−4, 6, 0, 0, 0, 2, 12}
	5	{−3, 2, 6, 0, 0, 2, 9}
	6	{−3, 2, 2, 6, 0, 2, 7}
	7	{−3, 1, 2, 2, 6, 3, 5}
1	0	{−10, 16, 0, 0, 0, 10, 0}
	1	{−6, 0, 16, 0, 0, 6, 0}
	2	{−4, 0, 0, 16, 0, 4, 0}
	3	{−2, 0, 0, 0, 16, 2, 0}
	4	{−10, 16, 0, 0, 0, 0, 10}
	5	{−6, 0, 16, 0, 0, 0, 6}
	6	{−4, 0, 0, 16, 0, 0, 4}
	7	{−2, 0, 0, 0, 16, 0, 2}
2	0	{−8, 8, 0, 0, 0, 16, 0}
	1	{−8, 0, 8, 0, 0, 16, 0}
	2	{−8, 0, 0, 8, 0, 16, 0}
	3	{−8, 0, 0, 0, 8, 16, 0}
	4	{−4, 4, 0, 0, 0, 0, 16}
	5	{−4, 0, 4, 0, 0, 0, 16}
	6	{−4, 0, 0, 4, 0, 0, 16}
	7	{−4, 0, 0, 0, 4, 0, 16}
3	0	{−2, 8, 0, 0, 0, 10, 0}
	1	{−1, 3, 8, 0, 0, 6, 0}
	2	{−1, 2, 3, 8, 0, 4, 0}
	3	{0, 1, 2, 3, 8, 2, 0}
	4	{−1, 4, 0, 0, 0, 3, 10}
	5	{−1, 3, 4, 0, 0, 4, 6}
	6	{−1, 2, 3, 4, 0, 4, 4}
	7	{−1, 2, 2, 3, 4, 3, 3}
4	0	{−12, 14, 0, 0, 0, 14, 0}
	1	{−10, 0, 14, 0, 0, 12, 0}
	2	{−9, 0, 0, 14, 0, 11, 0}
	3	{−8, 0, 0, 0, 14, 10, 0}
	4	{−10, 12, 0, 0, 0, 0, 14}
	5	{−9, 1, 12, 0, 0, 0, 12}
	6	{−8, 0, 0, 12, 0, 1, 11}
	7	{−7, 0, 0, 1, 12, 1, 9}

```
FOR i in 0 to 7:
  aux = 0
  FOR j in 0 to 6:
    aux = aux + FilterCoefficients[MODE][i][j] * L[j]
  END FOR
  Block[i] = (aux + 8) / 16
END FOR
```

Figure 10. Recursive-based-filtering base algorithm for a single 4x2 subblock. This process is repeated for every subblock inside the predicted block. Then, all predicted subblocks are copied accordingly to create the *Pred* array.

3.5. Chroma-from-Luma (CfL) Mode

The idea of predicting chrominance samples from reconstructed luminance samples was first proposed for the HEVC standard in [Chen 2011], but it was rejected, as it added considerable complexity to the decoder. An adapted version was later proposed again as part of the HEVC Range Extension in [Pu 2013]. The AV1 version of CfL is an improvement of the variants found in both Thor and Daala.

It is a chrominance-only predictor that works as shown in Figure 11. First, to predict two blocks of chrominance (Cb and Cr) samples, the reconstructed block of luminance (Y) samples must be subsampled to the same size of the chrominance blocks. Then, from each luminance sample, the average of all samples is subtracted, resulting in an AC Contribution block. Meanwhile, from the Cb and Cr chrominance samples, two regular DC predicted blocks are calculated. Finally, scaled versions of the AC Contribution of luminance samples is added to the DC predicted chrominance blocks. There is a scaling factor for each chrominance channel and both must be signaled in the bitstream, adding to the cost of this prediction mode.

Each of the scaling factors are defined as a 3-value symbol representing a sign, which can be either negative, zero or positive. Only if not zero, a 16-value symbol is used, representing a factor ranging from 0 to 2 with a step of 1/8. A scaling factor equal to zero for one of the chrominance planes leads to the luminance contribution being discarded, resulting in a standard chrominance DC prediction for that channel. Because of that, the bitstream structure does not allow both factors to be zero.

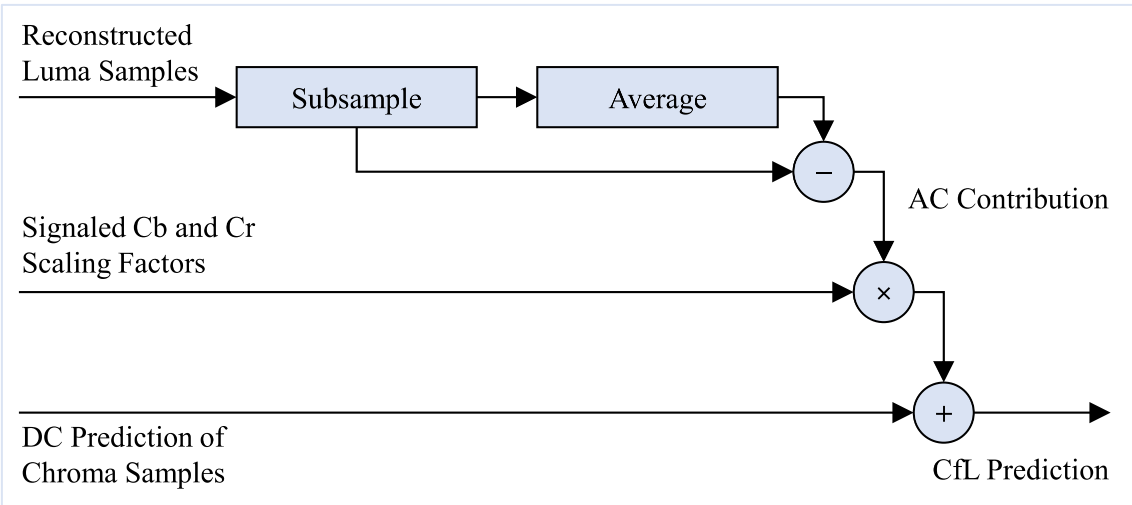

Figure 11. Illustration of the CfL prediction algorithm.

3.6. Directional Modes

There are eight base directional modes in AV1, called nominal angles, on which fine angle variations of −9, −6, −3, 0, +3, +6 and +9 degrees are introduced, totaling 56 possible angles. The nominal angles and each possible angle variation (*pAngle*) are listed in Table 4 and illustrated in Figure 12. Also, examples of predicted blocks for each of the eight nominal angles are shown in Figure 13.

All 56 prediction angles are calculated using a unified directional algorithm that links each predicted sample to a reference sub-sample position in *AboveRow* or *LeftCol*. Then, the two reference samples closest to the sub-sample position are interpolated using a 2-tap bilinear filter [Chen 2018].

Table 4. Angles supported by the AV1 directional intra prediction.

Intra mode (Nominal angle)	Possible values for *pAngle* (degrees)						
D45_PRED	36	39	42	**45**	48	51	54
D67_PRED	58	61	64	**67**	70	73	76
V_PRED	81	84	87	**90**	93	96	99
D113_PRED	104	107	110	**113**	116	119	121
D135_PRED	126	129	132	**135**	138	141	144
D157_PRED	148	151	154	**157**	160	163	166
H_PRED	171	174	177	**180**	183	186	189
D203_PRED	194	197	200	**203**	206	209	211

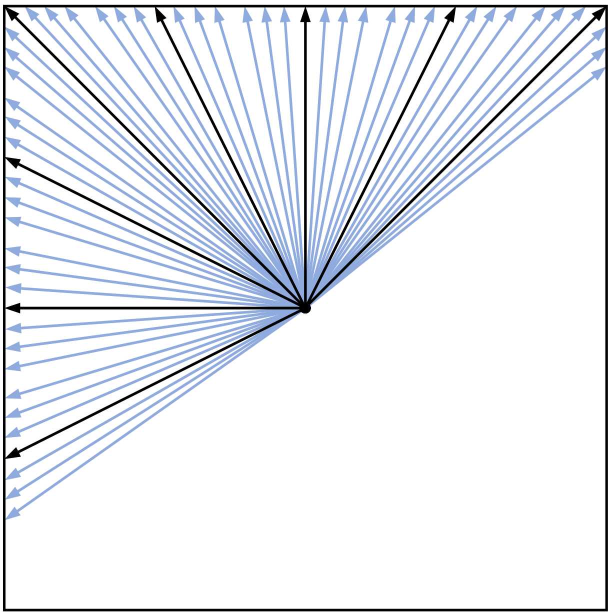

Figure 12. AV1 intra prediction nominal angles (black arrows) and respective offsets (blue arrows).

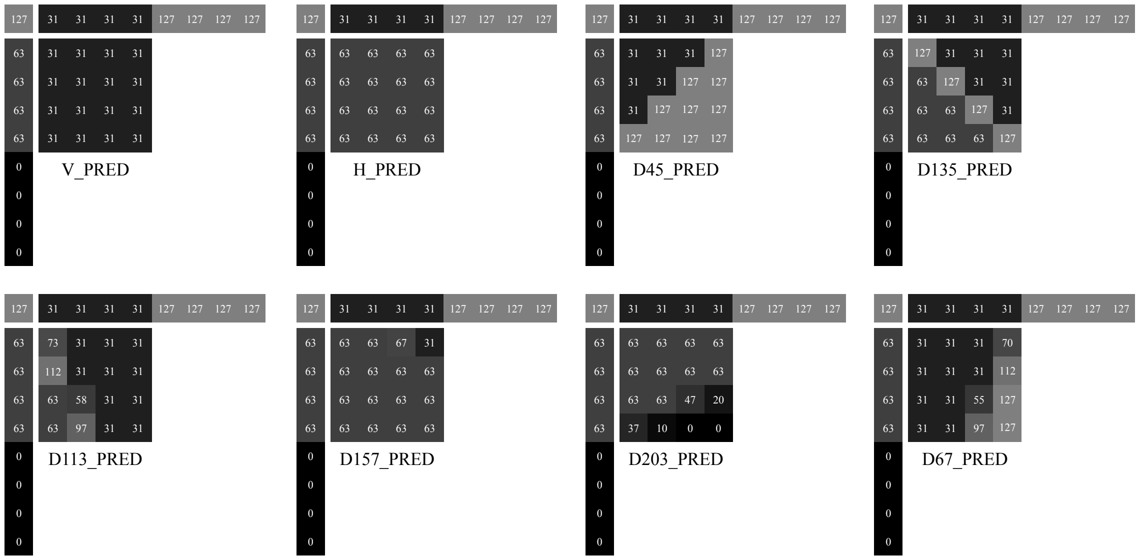

Figure 13. Directional predictions of the eight nominal angles for a 4x4 block.

If allowed by a flag in the sequence header, then a filtering and upscaling process can be applied to the reference samples of *AboveRow* and *LeftCol*, preceding the prediction process itself.

A wide set of conditions must be met for determining if a specific type of filter will be used or not. For a more detailed description, please refer to Rivaz (2018, pg. 233). Basically, this process can apply:

- A 3-tap filter to the *AboveRow*[−1] and *LeftCol*[−1] using the adjacent references stored in *AboveRow*[0] and *LeftCol*[0], depending on the values of *pAngle*, *width* and *height*;
- A 5-tap filter on the remaining samples of *AboveRow* and *LeftCol*, depending if the top and left neighboring transform blocks are available. There are three different set of weights for the 5-tap filter, the right one being selected based on values of *pAngle*, *width*, *height*, and also if the top and left neighbors were predicted using one of the three smooth predictors.

The upscaling process can be applied independently for each reference array (i.e. the references can be stretched only in one direction). For both arrays, the upscaling can only happen if $(width + height \leq S)$, where $S = 8$ if one of the neighbors were coded using one of the smooth predictors, otherwise $S = 16$. Additionally, the upscaling of *AboveRow* can only happen for *pAngle* values from 93 to 129 degrees inclusive, and for *LeftCol* it can only happen for values from 183 to 211 inclusive. When invoked, the upscaling process doubles the size of one or both arrays, filling each position between the original reference samples with a half-sample interpolated using a 4-tap filter.

The directional prediction has five different cases, depending on *pAngle*, which determines if the prediction will need one or both arrays of references samples. In any case, the process starts by setting a variable *dx* and a variable *dy*, as described by both Tables 5 and 6.

Table 5. Derivation of variables *dx* and *dy* (part 1).

Variable	Derivation Procedure
dx	If *pAngle* < 90, then *dx* is set according to Table 6 using *pAngle* as reference angle. Otherwise, if *pAngle* > 90 and *pAngle* < 180, then *dx* is set according to Table 6 using 180−*pAngle* as reference angle.
dy	If *pAngle* > 90 and *pAngle* < 180, then *dy* is set according to Table 6 using *pAngle*−90 as reference angle. Otherwise, if *pAngle* > 180, then *dy* is set according to Table 6 using 270−*pAngle* as reference angle.

Table 6. Derivation of variables *dx* and *dy* (part 2).

Reference Angle	Derived Angle
3	1023
6	547
9	372
14	273
17	215
20	178
23	151
26	132
29	116
32	102
36	90
39	80
42	71
45	64
48	57
51	51
54	45
58	40
61	35
64	31
67	27
70	23
73	19
76	15
81	11
84	7
87	3

The prediction itself is described in Figure 14, where *upsampleAbove* and *upsampleLeft* are, respectively, flags indicating if the reference arrays *AboveRow* and *LeftCol* are upscaled. Although omitted in the figure below, predicted samples must also be clipped according to the bit depth.

```
FOR i in 0 to height-1:
  FOR j in 0 to width-1:

    IF pAngle < 90:
      idx     = (i + 1) * dx
      base    = (idx >> (6 – upsampleAbove)) + (j << upsampleAbove)
      shift   = (idx << upsampleAbove) >> 1) & 31
      maxBase = (w+h-1) << upsampleAbove
      IF base < maxBase:
        Pred[i][j] = AboveRow[base] * (32 – shift) + AboveRow[base+1] * shift
        Pred[i][j] = (Pred[i][j] + 16) / 32
      ELSE:
        Pred[i][j] = AboveRow[maxBase]
      END IF

    ELSE IF pAngle > 90 AND pAngle < 180:
      idx  = (j << 6) – (i + 1) * dx
      base = idx >> (6 – upsampleAbove)
      IF base >= –(1 << upsampleAbove):
        shift = ((idx << upsampleAbove) >> 1) & 31
        Pred[i][j] = AboveRow[base] * (32 – shift) + AboveRow[base + 1] * shift
      ELSE:
        idx   = (i << 6) – (j + 1) * dy
        base  = idx >> (6 – upsampleLeft)
        shift = ((idx << upsampleLeft) >> 1) & 31
        Pred[i][j] = LeftCol[base] * (32 – shift) + LeftCol[base + 1] * shift
      END IF
      Pred[i][j] = (Pred[i][j] + 16) / 32

    ELSE IF pAngle > 180:
      idx     = (j + 1) * dy
      base    = (idx >> (6 – upsampleLeft)) + (i << upsampleLeft)
      shift   = (idx << upsampleLeft) >> 1) & 31
      Pred[i][j] = LeftCol[base] * (32 – shift) + LeftCol[base + 1] * shift
      Pred[i][j] = (Pred[i][j] + 16) / 32

    ELSE IF pAngle == 90:
      Pred[i][j] = AboveRow[j]

    ELSE IF pAngle == 180:
      Pred[i][j] = LeftCol[i]
    END IF

  END FOR
END FOR
```

Figure 14. Unified directional prediction algorithm for any supported *pAngle*.

3.7. Screen Content Coding (SCC) Modes

AV1 supports two different predictors for SCC: the Color Palette and the Intra Block Copy predictors.

The Color Palette predictor is beneficial when blocks can be approximated by a small number of unique colors. This mode is allowed only for blocks of size 8x8 or bigger. The bitstream structure requires an array representing a color palette of two to eight colors to be signaled and, also, a structure map, which is a 2D array filled with the indexes of the colors (according to the palette) to be used in the prediction. The encoder can explore different size of palettes and different colors to optimize the resulting rate-distortion cost. An 8x8 block approximately predicted by a palette of five colors is shown in Figure 15.

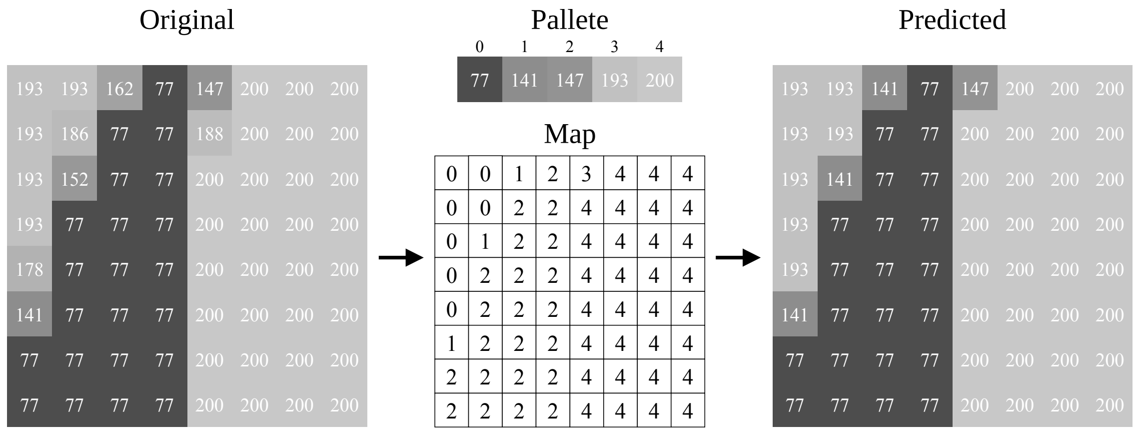

Figure 15. A hypothetical Palette prediction for an 8x8 block.

The Intra Block Copy predictor allows the intra coder to refer back to previously coded samples in the same frame the same way the inter coder refers back to previously coded samples in previously coded frames. It is very beneficial in screen content frames where many repeated textures and patterns are present. The location of the reference block is specified by a displacement vector in a way similar to motion vector compression in motion compensation. Displacement vectors are integer for the luminance channel, and may be fractional for chrominance planes, in which case bilinear filtering is applied [Chen 2018]. This mode is only allowed for intra frames, to which conventional inter prediction does not happen.

4. Intra Prediction on VP9 and HEVC

This section presents briefly the intra prediction tools of the VP9 (predecessor of AV1) and HEVC (direct competitor of AV1) video formats. However, the objective of the following discussion is not to compare the intra prediction performance among different video formats, as that cannot be done effectively without also comparing transform coding, entropy coding and bitstream structure.

4.1. VP9 Tools

In VP9, the SB maximum size is 64x64 pixels. The format allows breakdown using a recursive decomposition all the way down to 4x4 blocks [Mukherjee 2013], as shown in Figure 16.

The transform coding for intra blocks is based on 2D hybrid transforms. For transform sizes of 4x4, 8x8 and 16x16, the hybrid transforms supported are the four combinations of 1D DCT and 1D ADST. For the transform size of 32x32, only the 2D DCT is supported. Additionally, a 4x4 Walsh Hadamard Transform (WHT) is used only when lossless encoding is desired [Mukherjee 2013].

There are two non-directional and eight directional intra modes in the VP9 specification. The non-directional are the DC mode (similar to the AV1 algorithm) and the TM mode. The TM mode, which is not present in AV1, is simply the use of the clipped *base* value of the AV1 Paeth mode (section 3.2) as predicted sample. The directional modes correspond approximately to angles 27, 45, 63, 117, 135 and 153, plus the horizontal and vertical modes [Mukherjee 2013].

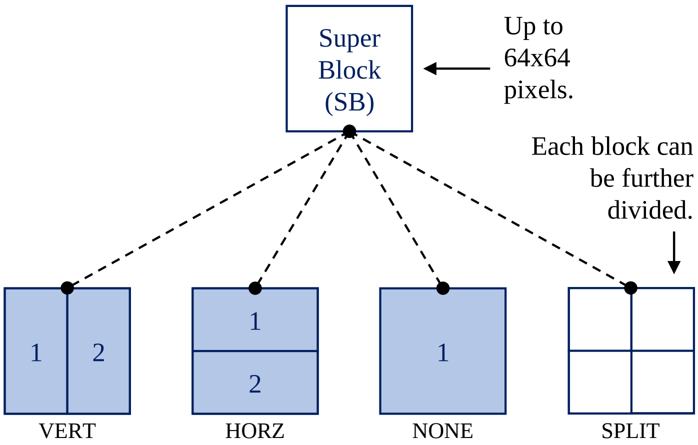

Figure 16. VP9 4-way block partition tree structure.

4.2. HEVC Tools

The core of the coding layer in the HEVC predecessors was the macroblock of size 16x16, whereas the analogous structure in HEVC is the Coding Tree Unit (CTU), which the encoder can define as 16x16, 32x32 or 64x64 [Sullivan 2012]. The CTUs can be recursively divided into four symmetrical blocks of the same size, called Coding Units (CUs). Intra predicted CUs can have a minimum size of 4x4.

Then, the CUs can be further divided into smaller Transform Units (TUs). For TUs of size 4x4, 8x8, 16x16 and 32x32, the 2D DCT is mainly used. Additionally, for the luminance samples of an intra TU of size 4x4, a 2D DST is also supported [Budagavi 2013].

There are two non-directional and 33 directional intra modes in the HEVC standard. The non-directional are the DC mode and Planar mode [Lainema 2012]. The DC mode in HEVC differs from the AV1 version by always taking the average of all reference samples, even if those are not available, in which case the closest available references are replicated by a pre-prediction step. The Planar mode is an algorithm with a similar behavior of the AV1 Smooth mode, although it does not use the analogue versions of Smooth Vertical and Smooth Horizontal as prediction modes.

The directional modes are composed by 33 prediction directions at 1/32 sample accuracy, as illustrated in Figure 17. The direction distribution considers the fact that, in natural imagery, horizontal and vertical patterns typically occur more frequently than other patterns, thus small displacement parameters are used for modes close to horizontal and vertical directions to take advantage of that phenomenon [Lainema 2012]. Each predicted sample is obtained by projecting its location to a reference row of column of references, then by performing a linear interpolation at 1/32 sample accuracy using the two closes reference samples [Lainema 2012].

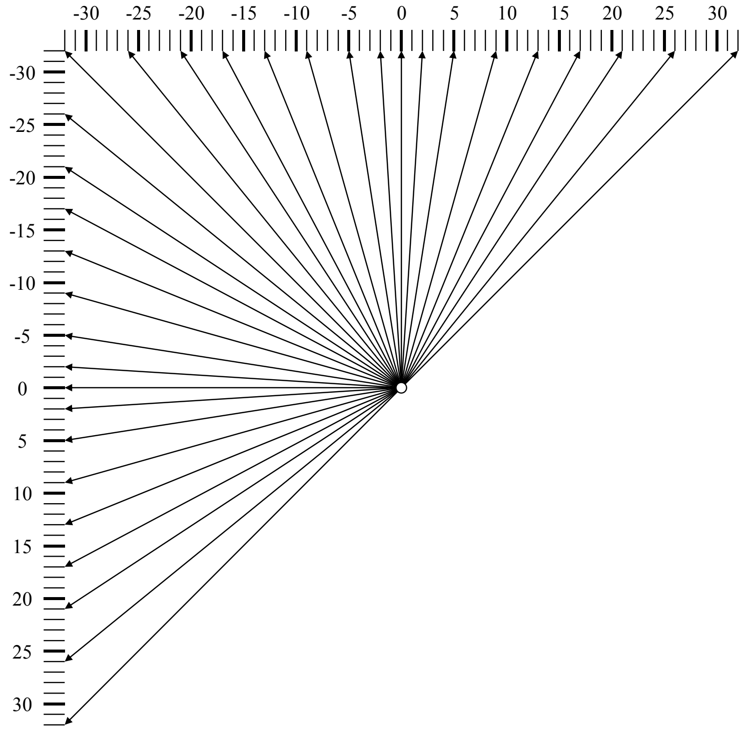

Figure 17. HEVC intra prediction directions and the associated displacement parameters, where the numbers refer to the displacement from the horizontal and vertical directionalities as 1/32 sample fractions [ISO/IEC 2013].

5. Our Contributions and Future Works

There are several published works related to the intra prediction of previous codecs available in the literature, such as [Fang 2016; Pastuszak 2016; Corrêa 2017; Min 2017; Palomino 2012; Zhou 2013]. However, these solutions are not compliant with the AV1 specification, mainly because: (a) both Paeth and Recursive-based-filtering predictors are novel modes which were not present in any previous codec, (b) although the Smooth predictor can be considered a variation of the HEVC Planar mode, the coefficients applied are different, and (c) the AV1 block partitioning structure is very different even from the most advanced previous codecs. This way, as more advanced video formats are developed, specialized hardware architectures become of paramount importance to enable video-based applications.

Since the AV1 is a very recent and important subject for both industry and academy, our recent contributions on the topic have been well received in Circuits and Systems (CAS) conferences such as LASCAS 2019 and ISCAS 2019, which happened on Sapporo–Japan and Armenia–Colombia, respectively.

Firstly, the LASCAS 2019 paper named "*A High Throughput Hardware Architecture Targeting the AV1 Paeth Intra Predictor*" [Corrêa 2019a], was developed and then submitted for review just three months after the AV1 specification release date. On this paper, we published a hardware design specifically for the Paeth predictor, which is one among ten other predictors that compose the AV1 non-directional intra prediction module. This was the first CAS-related work published on the subject.

Secondly, the ISCAS 2019 paper named "*High Throughput Hardware Design for*

AV1 Paeth and Smooth Intra Modes" [Corrêa 2019b], was submitted for review five months after the AV1 release date. On this paper, we published a more energy efficient design for the Paeth predictor and also the design for all three Smooth predictors.

Finally, after receiving an invitation from TCAS-I to publish an extended version of [Corrêa 2019a], we developed and submitted a paper describing a complete AV1 non-directional intra prediction module, including not only the abovementioned predictors, but also the DC and Recursive-based-filtering (modes 0 to 4) predictors. This paper is still under the reviewing process.

As future works, we intend not only to continue working on hardware designs for intra prediction, but also on algorithms to reduce encoding complexity.

On hardware design, our next milestone is to develop a complete AV1 directional intra prediction unit. After that, developing hardware designs for the Versatile Video Coding (VVC) standard, which is a work in progress for a HEVC successor, is also an open challenge for us.

On complexity reduction, it is important to mention that the increase in both the number of intra prediction modes and possible block partitions is notable in AV1, thus making its encoder much more complex than the VP9 and HEVC counterparts. Therefore, a local fast mode decision for the intra prediction is very important for both software and hardware solutions. In this field, we already developed and implemented on the reference HEVC encoder a fast algorithm for local mode decision based on the same direction detection technique used by the AV1 Constrained Directional Enhancement Filter (CDEF) [Midtskogen 2018]. We were able to achieve very competitive results in encoding time and rate-distortion and a paper names "*A Fast Local Mode Decision for the HEVC Intra Prediction Based on Direction Detection*" [Corrêa 2019c] was published on the conference EUSIPCO 2019, that happened in A Coruña–Spain in September 2019. As future work, we intend to adapt this algorithm to the AV1 intra encoding tools.

6. Conclusions

This monograph presented the block structure, transform coding and intra coding of the AV1 format, along with a brief comparison with other codecs. The current contributions of the author in this theme were also briefly described.

The AV1 is the next generation free video format, which brings many innovations, particularly on the intra prediction module when compared to both VP9 and HEVC.

To make high definition video coding feasible in real time, such as the UHD 8K resolution at 120 frames per second, many research challenges on intra prediction must be explored in both hardware development and software algorithms.

References

Bankoski, J.; et al. "RFC 6386 VP8 Data Format and Decoding Guide," 2011. Available: https://datatracker.ietf.org/doc/rfc6386/

Bjøntegaard, G.; Davies, T.; Fuldseth, A.; Midtskogen, S. "The Thor Video Codec," Data Compression Conference (DCC), Snowbird, USA, 2016.

Budagavi, M.; et al. "Core Transform Design in the High Efficiency Video Coding (HEVC) Standard," IEEE Journal of Selected Topics in Signal Processing, v.7, n.6, p.1029-1041, 2013.

Chen, J.; et al. “CE6.a.4: Chroma intra prediction by reconstructed luma samples,” JCTVC-E266 meeting document, Geneva, Switzerland, 2011.

Chen, Y.; et al. “An Overview of Core Coding Tools in the AV1 Video Codec,” Picture Coding Symposium, San Francisco, USA, 2018.

Cisco Systems, “Cisco Visual Networking Index: Forecast and Trends, 2017–2022 White Paper,” 2017. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.pdf

Corrêa, M.; Zatt, B.; Porto, M.; Agostini, L. “High-throughput HEVC intrapicture prediction hardware design targeting UHD 8K videos,” IEEE International Symposium on Circuits and Systems (ISCAS), Baltimore, USA, 2017.

Corrêa, M.; Waskow, B.; Goebel, J.; Palomino, D.; Corrêa, G.; Agostini, L. “A High Throughput Hardware Architecture Targeting the AV1 Paeth Intra Predictor,” IEEE Latin American Symposium on Circuits & Systems (LASCAS), Armenia, Colombia, 2019.

Corrêa, M.; Waskow, B.; Zatt, B.; Palomino, D.; Corrêa, G.; Agostini, L. “High Throughput Hardware Design for AV1 Paeth and Smooth Intra Modes,” IEEE International Symposium on Circuits and Systems (ISCAS), Sapporo, Japan, 2019.

Corrêa, M.; Zatt, B.; Palomino, D.; Corrêa, G.; Agostini, L. “A Fast Local Mode Decision for the HEVC Intra Prediction Based on Direction Detection,” European Signal Processing Conference (EUSIPCO), A Coruña, Spain, 2019.

Fang, H.; Chen, H.; Chang, T. “Fast intra prediction algorithm and design for high efficiency video coding,” IEEE International Symposium on Circuits and Systems (ISCAS), Montreal, Canada, 2016.

Grange, A.; Rivaz, P.; Hunt, J. “VP9 Bitstream & Decoding Process Specification,” 2013. Available: https://www.webmproject.org/vp9/

Guo, L.; et al. “Color palette for screen content coding,” IEEE International Conference on Image Processing (ICIP), Paris, France, 2014.

ISO/IEC. Information technology: High efficiency coding and media delivery in heterogeneous environments – Part 2: High efficiency video coding, ISO/IEC 23008-2. 2013. 296p.

Lainema, J.; et al. “Intra Coding of the HEVC Standard,” IEEE Transactions on Circuits and Systems for Video Technology, v.22, n.12, p.1792-1801, 2012.

Li, J.; et al. “Intra Block Copy for Screen Content in the Emerging AV1 Video Codec,” Data Compression Conference (DCC), Snowbird, USA, 2018.

Midtskogen, S.; Valin, J. “The AV1 Constrained Directional Enhancement Filter (CDEF),” IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, Canada, 2018.

Min, B.; Xu, R.; Cheung, R. “A Fully Pipelined Hardware Architecture for Intra Prediction of HEVC,” IEEE Transactions on Circuits and Systems for Video Technology, v.27, n.12, p.2702-2713, 2017.

Mukherjee, D.; et al. “A Technical Overview of VP9 - The Latest Open-Source Video Codec,” SMPTE Annual Technical Conference & Exhibition, Hollywood, USA, 2013.

Palomino, D.; et al. “A memory aware and multiplierless VLSI architecture for the complete intra prediction of the HEVC emerging standard,” IEEE International Conference on Image Processing (ICIP), Orlando, USA, 2012.

Parker, S.; et al. “On Transform Coding Tools Under Development for VP10,” SPIE Optical

Engineering + Applications, San Diego, USA, 2016.

Pastuszak, G.; Abramowski, A. "Algorithm and architecture design of the H.265/HEVC intra encoder," IEEE Transactions on Circuits and Systems for Video Technology, v.26, n.1, p.210-222, 2016.

Pu, W.; et al. "Non-RCE1: Inter Color Component Residual Prediction," JCTVC-N0266 meeting document, Vienna, Austria, 2013.

Rivaz, P.; Haughton, J. "AV1 Bitstream & Decoding Process Specification," 2018. Available: https://aomedia.org/av1-bitstream-and-decoding-process-specification/

Rosenberg, J. "World, Meet Thor – a Project to Hammer Out a Royalty Free Video Codec," 2015. Available: https://blogs.cisco.com/collaboration/world-meet-thor-a-project-to-hammer-out-a-royalty-free-video-codec/

Sullivan, G. J.; Ohm, J.; Han, W.; Wiegand, T. "Overview of the High Efficiency Video Coding (HEVC) Standard," IEEE Transactions on Circuits and Systems for Video Technology, v.22, n.12, p.1649-1668, 2012.

Trudeau, L.; Egge, N.; Barr, D. "Predicting Chroma from Luma in AV1," Data Compression Conference (DCC), Snowbird, USA, 2018.

Valin, J.; et al. "Daala: Building a next-generation video codec from unconventional technology," IEEE 18th International Workshop on Multimedia Signal Processing (MMSP), Montreal, Canada, 2016.

Zhou, N,; Ding, D.; Yu, L. "On hardware architecture and processing order of HEVC intra prediction module," Picture Coding Symposium (PCS), San Jose, USA, 2013.